CP 111 Computer Programming

2/2554 Section B01/B02

Lab 10

หมายเหตุ การเขียนโปรแกรมที่คำนวณกับตัวเลขที่มีจุดทศนิยม ให้ประกาศตัวแปรที่เกี่ยวข้องเป็นแบบ `double` (Double-precision floating point)

1. Problem name: `Tower_of_Hanoi`

จงเขียนโปรแกรมโดยใช้เทคนิค Recursion เพื่อแสดงคำตอบหรือผลเฉลยของปัญหา Tower of Hanoi โดยกำหนดให้โปรแกรมรับข้อมูลนำเข้าเป็นพารามิเตอร์ของปัญหา ได้แก่

n = จำนวน disk ที่จะเคลื่อนย้าย (Disk 1=เล็กสุด และ Disk n=ใหญ่สุด)
startPost = หมายเลขเสาเริ่มต้น
endPost = หมายเลขเสาปลายทาง
tempPost = หมายเลขเสาตัวช่วย

โดยโปรแกรมจะแสดงขั้นตอนการย้าย Disk เป็นไปตามกฎ ในรูปแบบบรรทัดละ 1 ขั้นตอน แต่ละขั้นตอน (บรรทัด) แสดง

<หมายเลข Disk> <หมายเลขเสาเริ่มต้น> <หมายเลขเสาปลายทาง>

ตัวอย่างการทำงานของโปรแกรม

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)

`3 1 3 2`	`1 1 3`
	`2 1 2`
	`1 3 2`
	`3 1 3`
	`1 2 1`
	`2 2 3`
	`1 1 3`

Note: ตัวอย่างข้างบนผู้ใช้ป้อนค่าเพื่อสั่งให้หาผลเฉลยของการเคลื่อนย้าย n=3 disk ที่อยู่ที่ startPost=1 ไปยังเสา endPost=3 โดยใช้เสา tempPost=2 เป็นตัวช่วย โดยสองแรกขั้นตอนของผลเฉลยคือ 1 1 3 และ 2 1 2 หมายถึงให้เริ่มต้นย้าย disk 1 จากเสา 1 ไป 3 ตามด้วยย้าย disk 2 จากเสา 1 ไป 2

2. Problem name: `Recur_Binary_Search`

จงเขียนโปรแกรมโดยใช้เทคนิค Recursion เพื่อหาว่าข้อมูลที่ต้องการค้นหาสมมติเรียกว่า b อยู่ในอินพุตอาร์เรย์ (สมมติเรียกว่า A) หรือไม่ ถ้ามีข้อมูลอยู่ให้แสดง index ของอาร์เรย์ดังกล่าว แต่ถ้าไม่มีข้อมูลอยู่ให้แสดงค่า -1

กำหนดให้ข้อมูลนำเข้าในบรรทัดแรกคือ n ซึ่งแทนจำนวน element ในอาร์เรย์ หลังจากนั้น n บรรทัดถัดมา จะเป็นข้อมูลแบบ int แทนข้อมูลในแต่ละ element ของอาร์เรย์ A หลังจากนั้นในบรรทัดสุดท้ายจะเป็นข้อมูล b

กำหนดให้ข้อมูลใน A มีไม่เกิน 100 ตัวและ*ถูกป้อนเข้ามาโดยเรียงจากมากไปหาน้อยแล้ว*

ตัวอย่างการทำงานของโปรแกรม

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)
11 926 777 392 387 153 110 109 49 37 32 12 110	5
11 926 777 392 387 153 110 109 49 37 32 12 50	-1

Hint

```
#include <stdio.h>

int BinarySearch(int b, int *A, int start, int end);

int main()
{
int data[100],i,n,b,retval;

      // Read n
      scanf(           );

      // read data[0]... data[n-1]
      for(             )
        scanf(                 );

      // read b
      scanf(           );

      retval=BinarySearch(                 );

      printf("%d\n",retval);

      return 0;
}
int BinarySearch(int b, int *A, int start, int end)
{



}
```

3. Problem name: `Iter_Binary_Search`

จงเขียนโปรแกรมโดยใช้ Iteration หรือ ลูป for, while, do-while เพื่อหาว่าข้อมูลที่ต้องการค้นหาสมมติเรียกว่า b อยู่ในอินพุตอาร์เรย์ (สมมติเรียกว่า A) หรือไม่ ถ้ามีข้อมูลอยู่ให้แสดง index ของอาร์เรย์ดังกล่าว แต่ถ้าไม่มีข้อมูลอยู่ให้แสดงค่า -1

กำหนดให้ข้อมูลนำเข้าในบรรทัดแรกคือ n ซึ่งแทนจำนวน element ในอาร์เรย์ หลังจากนั้น n บรรทัดถัดมา จะเป็นข้อมูลแบบ int แทนข้อมูลในแต่ละ element ของอาร์เรย์ A หลังจากนั้นในบรรทัดสุดท้ายจะเป็นข้อมูล b

กำหนดให้ข้อมูลใน A มีไม่เกิน 100 ตัวและ*ถูกป้อนเข้ามาโดยเรียงจากมากไปหาน้อยแล้ว*

ตัวอย่างการทำงานของโปรแกรม

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)
11 926 777 392 387 153 110 109 49 37 32 12 110	5
11 926 777 392 387 153 110 109 49 37 32 12 50	-1

Hint:

```
#include <stdio.h>

int Iter_Binary_Search(int *A,int n, int b);

int main()
{
int data[100],i,n,b,retval;

      scanf("%d",&n);
      for(i=0;i<n;i++)
        scanf("%d",&data[i]);

       scanf("%d",&b);

       retval=Iter_Binary_Search(             );

       printf("%d\n",retval);

       return 0;
}
int Iter_Binary_Search(int *A,int n, int b)
{
int start,mid,end;

         start=
         end=
         while(1)
         {
             if(               )
                 return(-1);

             mid=

             if(               )
             {
                return(       );
             }
             else if(                  )
             {

                start=
                end=
             }
             else
             {
               start=
               end=
             }
        }
}
```

4. Problem name: `Recur_Occurrences`

จงใช้เทคนิค Recursion เพื่อเขียนโปรแกรมทำการนับจำนวนตัวอักษรหนึ่งๆปรากฏอยู่ในสตริง โดยโปรแกรมจะรับข้อมูลนำเข้าเป็นสตริงและตัวอักษรที่ต้องการค้นหา หลังจากนั้นโปรแกรมจะแสดงจนวนครั้งที่ตัวอักษรดังกล่าวได้ปรากฏในสตริง

กำหนดให้เขียนโค้ดในในฟังก์ชันชื่อ `Occurrences` เพื่อทำการนับจำนวนตัวอักษรที่ระบุโดยอาร์กิวเมนต์ `ch` ที่ปรากฏอยู่ในข้อความหรือสตริงที่ระบุโดยอาร์กิวเมนต์ `str` และคืนค่าจำนวนตัวอักษรที่นับได้นี้กลับไปยัง Caller function

ข้อกำหนด: ให้การเขียนโค้ดในฟังก์ชันนี้ให้เขียนโดยใช้เทคนิค Recursion เท่านั้น นั้นคือฟังก์ชัน `Occurrences` ที่เสร็จสมบูรณ์ จะต้องเป็นฟังก์ชันแบบเรียกตัวเอง (Recursive function) เท่านั้น

ตัวอย่างการทำงานของโปรแกรม

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)
CP121/CP111 1	5
praditm@swu.ac.th P	0
PRADIT MITTRAPIYANURUK R	3

Hint

```
#include <stdio.h>
int Occurrences(char *,char);
void main()
{
char buff[100];
char ch;
int count;
      gets(buff);
      scanf("%c",&ch);
      count=Occurrences(buff,ch);
      printf("%d\n",count);
}
int Occurrences(char *str,char ch)
{
      if(                        ) // base case
            return 0;
      else if(                   )
            return(1+                    );
      else
            return(                       );

}
```

5. Problem name: `Recur_Minimum`

จงใช้เทคนิค Recursion เพื่อเขียนโปรแกรมทำการหาข้อมูลที่น้อยที่สุดในอาร์เรย์ โดยกำหนดให้ข้อมูลนำเข้าในบรรทัดแรกคือ n ซึ่งแทนจำนวน element ในอาร์เรย์ หลังจากนั้น n บรรทัดถัดมา จะเป็นข้อมูลแบบ int แทนข้อมูลในแต่ละ element ของอาร์เรย์ กำหนดให้ข้อมูลในอาร์เรย์มีไม่เกิน 100 ตัว และ ให้เขียนและเรียกใช้ฟังก์ชัน Recur_Minimum ดังนิยามข้างล่าง

```
int Recur_Minimum(int *A, int start, int end)
```

ซึ่งฟังก์ชันนี้จะทำหน้าที่ในการหาค่าน้อยสุดในอาร์เรย์ A ตั้งแต่ element ที่ start ถึง end (A[start]..A[end])

ตัวอย่างการทำงานของโปรแกรม

ข้อมูลนำเข้า (Input)	ข้อมูลส่งออก (Output)
5 325 732 -520 983 881	5

Hint:

```
#include <stdio.h>
int Recur_Minimum(int *A, int start, int end);
int main()
{
int data[100],i,n,b,min;
        scanf("%d",&n);
        for(i=0;i<n;i++)
           scanf("%d",&data[i]);
        min=Recur_Minimum(                    );
        printf("%d\n",min);
        return 0;
}
int Recur_Minimum(int *A, int start, int end)
{
int x;
      if(start==end)
            return(                    );
      else{
            x=Recur_Minimum(                    );
            if(x<A[end])
                  return(                 );
            else
                  return(                 );
      }
}
```